

KEMI-TORNIO UNIVERSITY OF APPLIED SCIENCES

Mobile Programming with QT

Python Programming for Maemo/MeeGo Platform

Mobile Programming Project

Zhou Quan

Thesis of Information Technology Degree Program
Specialization
Bachelor of Engineering

KEMI 2011

PREFACE

Firstly, I would like to thank my instructor Mr. Teppo Aalto. I could not finish this thesis without his help.

Secondly, I would like to thank my parents; they always support me during my studies. I would like to thank all the teachers and staff of Kemi-Tornio University of Applied Sciences. Because of their hard work, I have a valuable experience of study in Finland during these years. It will be a great pleasure in my life.

ABSTRACT

Kemi-Tornio University of Applied Sciences, Technology	
Degree Programme	Information Technology
Name	Zhou Quan
Title	Python Programming for Maemo/MeeGo Platform
Type of Study	Bachelor's Thesis
Date	21 September 2011
Pages	30 + 5 appendices
Instructor	Teppo Aalto

Mobile devices already became one of the most necessary things nowadays. There are several mobile platforms in the market like Symbian S40, Symbian S60, Symbian Anna, Symbian Bella, IOS, Android, Maemo, MeeGo, and so on. Until now, Symbian series, IOS, and Android are the main platforms for mobile devices. The Android platform uses multi-brand method to extend the market. IOS is used on the product of Apple Company, but the number of the users is growing every day. Maemo is not a new platform. But it was designed to be used in Linux kernel. That makes this platform a professional platform. However, because of the open source feature, there are lots of homebrew applications can work on it.

Qt framework is supported by Maemo platform and that gives possibilities for the developing work. MeeGo platform is the next generation of Maemo, it has been published by Nokia this year. That means the Linux Mobile platform will become more popular in the future. As to the situation of Android right now, it is not only be used on mobile phones, but also on some micro computers for using quick start system.

In this study, a method of the developing work in Qt framework and using Python language is discussed. Qt framework and Python language are both supported by Maemo and MeeGo platforms. The product can work not only on Maemo/MeeGo platform, but also on Symbian series platforms directly. That means the idea of real cross-platforms can be implemented.

Keywords: Maemo, MeeGo, Qt framework, Python, PyQt4.

TABLE OF CONTENTS

1. INTRODUCTION	1
2. ENVIRONMENT	2
2.1. Python	2
2.1.1. Features	2
2.1.2. Perl & Python	4
2.2. Maemo/MeeGo	4
2.3. Eric4	5
2.4. Qt & PyQt	5
3. INSTALLATION OF ENVIRONMENT	7
3.1. Software requirement	7
3.1.1. Python	7
3.1.2. PyQt4	8
3.1.3. Eric 4	8
3.2. The steps of installation	9
3.3. Hello World Application	11
4. MINI WEB BROWSER	15
4.1. Overview	15
4.2. Implementation	15
5. ANALYSIS OF THE EXAMPLES	19
5.1. Basic input dialog	19
5.1.1. Function of "slotName()"	19
5.1.2. Function of "solotSex()"	20
5.1.3. The function of "slotAge()"	21
5.1.4. Function of "solotStature()"	21
5.2. Progress Bar	22
5.2.1. Using progress bar	23
5.2.2. Using progress dialog	23
6. PACKAGE	25
7. CONCLUTION	28
8. REFERENCES	29

EXPLANATION OF CHARCTERS AND ABBREVIATIONS

OS	Operating System
FLOSS	Free and Open Source Software
CPU	Central processing unit
IDE	Integrated Development Environment
SDK	Software Development Kit
URL	Uniform Resource Locator
CGI	Common Gateway Interface
FTP	File Transfer Protocol
XML	Extensible Markup Language
HTML	Hyper Text Markup Language
WAV	Waveform audio format
CPAN	Comprehensive Perl Archive Network

1. INTRODUCTION

The mobile devices are very popular nowadays. From the marketing success of iPhone and iPad, it is easy to find the factor why mobile products have huge market. That is the huge number of applications. There are 100,000 applications in App Store for iPhone and iPad until now. The programming method of applications for mobile devices is discussed in this work.

Python language is a programming language without a long history, but it is powerful. The applications which are developed by Python language can work everywhere without any changes. That means the applications are not only used on mobile devices, but also on personal computers. This feature could make the products to get larger market.

MeeGo is an operating system. It is based on Linux kernel. It was developed by the cooperation of Nokia and Intel. It is the next generation of Maemo platform. The developing work of MeeGo uses Qt framework. Several programming languages including Python and C++ are supported by Qt framework.

The background and the basic information of Python language, Qt framework and Maemo/MeeGo operating system are introduced in chapter 2. The installation method and steps are introduced in chapter 3. That is the way of building developing environment for this work. The steps of building “Hello world” testing application are introduced also in chapter 3. Example project building steps are introduced in chapter 4. For the purpose of the code analysis, two basic examples are introduced in chapter 5.

2. ENVIRONMENT

Before this work, learning the language and the platforms is necessary. The Python language and Maemo/MeeGo platform are introduced in this part.

2.1. Python

Python is an open source programming language. Compared to Perl, Ruby, Scheme, and Java, it seems more powerful and clear. It is a programming language which stands for a concise idea. To read a successful Python coding file, it feels like reading English directly, although this “English” has the stringent requirement. The most positive feature is that it makes the developer pay more attention to solving the problem, instead of wasting time on understanding the language. /1/ /2/ /3/

2.1.1. Features

Free and open source

Python is the member of FLOSS. That means, the users own full rights on the applications for free, including publish the copy, read the source code, edit the source code, and use some parts in new free applications. FLOSS is an idea of sharing the knowledge for the team. That is the reason of Python is so outstanding. /2/

High level language

The controlling of the processes and memories is the other feature of Python. That means, the basic details such as the application memory management methods are not considered during the development of the work.

Portability

Because of the nature of open source, Python language has already been transplanted on several platforms. If the devices have the same hardware, the application could be used on all platforms without any changes.

The platforms including: Linux, Windows, FreeBSD, Macintosh, Solaris, Amiga, BeOS, Palm OS, QNX, VMS, Psion, Acom RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE, PocketPC and so on./2/ /3/

Explanatory

When the application which developed by basic languages running, the sources files are translated to the computer languages (binary code). The connection or the reproduced software load the application from the hard disk to the memory and it is processed by CPU.

The application developed by Python language can run without being translated to the binary code. The CPU can run the application from the source files directly. Inside the computer, the Python translator could translate the source files to byte code intermediate form. After that, byte code intermediate form will be translated to computer language at the same time of running. Because the developer does not mind in method of compile the application and the connection of library correct or not, Python is easier than other languages. The user needs to copy the application to other computer and it runs well. This makes the Python application more portable. /2/ /3/

Object Oriented

Python is supported by Procedural programming and Object Oriented programming. In the process of Procedural programming, the application is built up by procedurals or some reusable code functions. In Object Oriented programming, the application is built up by data and features. Compare with other languages such as C++ language and Java language, Python comes into use realize the Object Oriented Programming powerful. /2/ /3/

Scalable

Either any key code is considered to run faster, or any function is considered to publish. These parts in C or C++ language could be programmed and they are used in Python application. /2/

Embeddability

The Python application in C/C++ application could be embedded. A scripting feature is supported to use in this way. /2/

Large library

The standard library of Python is really large. It will help the programmer do all kinds of work, including regular expression, document generation, unit testing, process, database, web browser, CGI, FTP, Email, XML, XML-RPC, HTML, WAV file, password system, GUI, TK and other operations about the system. These all functions are useable when Python is installed. It is called all functions idea of Python.

Except the standard library, there are lots of high quality libraries, such as wxPython, Teisted and Python Image library and so on. /3/

2.1.2. Perl & Python

Perl is another very popular open source programming language. To build a small application, Perl is a good choice. It can complete the small application, "Done the work". But when the application is large, Perl seems unpractical.

Compared with Perl, Python is easier, clearer, and easier to use. This makes it easier to understand, and easier to maintain. The most important advantage of Perl is that it has a large CPAN library. As this name, it has a large Perl plug-in library. It could do almost everything on computer. Perl has more plugs than Python, because Perl has a longer history.

Otherwise, the latest Parrot simulator is that it could run new Perl 6 also Python and other explanation languages like Ruby, PHP and TCL and so on. It means that the all Perl plugs will be used on Python in the future.

2.2. Maemo/MeeGo

Maemo

Maemo is a platform which is based on GNU Linux operating system and GNOME desktop. A functional and easy developing environment is supported by it. Using Meamo platform could easily be moved to existing mobile desktop solutions./9/ /10/

Maemo platform was first published in 2005. It is one of the top kernels for high level product of Nokia. Right now, Maemo is the base of several OS which are used on Internet Tablet. These micro computer mobile devices are designed for professional user, including vertical areas of small business users. The products include Nokia N800, N810 and N900. /9/

MeeGo/Harmattan

MeeGo is the new generation of Maemo platform for mobile devices. Both of these are based on Linux. The applications which are built on the Moblin or Maemo are compatible with the MeeGo. It is an open source Linux project which is managed by Linux Foundation. Therefore, the mobile developers will be advanced from Linux's free character greatly./1/ /9/ /10/

MeeGo 1.2 Harmattan by Nokia is a mature, feature-rich software platform designed to fit Nokia's latest cutting-edge device. The developers are allowed to create great-looking application easily by using the state-of-the-art Qt application framework and Qt SDK. The developers install Harmattan target for extending Qt SDK's capabilities to cover the Harmattan platform. /1/ /9/ /10/ /11/

2.3. Eric4

Eric is a full featured Python and Ruby editor and IDE which written in Python. It is based on the cross platform Qt GUI toolkit to integrate the highly flexible Scintilla editor control. It is considered like a quick and direct editor as well as a professional project management tool.

Eric4 includes a plug-in system. It allows easy extension of the IDE functionality with plug-in downloadable from the net. The current stable version is Eric4 based on Qt4 and Python 2 and eric5 based on Python 3 and Qt4. /14 /

Eric4 can be used in MS Windows X86 and X64, and also in Linux kernel operating systems (Ubuntu, Fedora). Because it is based on Qt GUI toolkit, the user interface design could be much easier than other tools for Python Development. The method and details of using Qt Design toolkit to design the user interface have been introduced in Chapter 4.

2.4. Qt & PyQt

Qt toolkit is a cross-platform GUI toolkit. It is widely used in the developing work of Windows, Linux, Mac OSX and lots of mobile devices. Qt has good framework on Object Oriented Programming, clear documents and intuitive API. /15/

Normally, the choosing of GUI for developing work is always a trouble to the developer. The developers who want to use Python language have a lot of choices. That is because the quality of GUI application for Python language is huge. In addition, they have their own advantages and disadvantages. Some of them have faster speed to run, some have a smaller storage, some are easier to install, and some are better on cross-platform. Different library has its own license. /15/

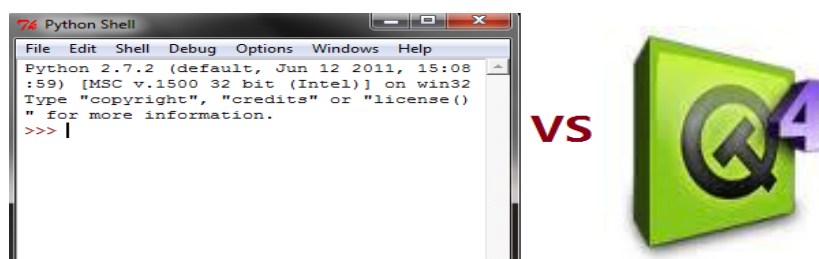


Fig.1. Tkinter & PyQt4 /2.4, page 5./

For the developer of Python, default GUI is Tk (linked through Tkinter). Tkinter and default IDE is developed by the founder of Python and that is the reason for the selection. They became the most default choices of the published Python versions.

Tkinter has been discussed in the standard Python document, but it has not been linked to any GUI. At least, if Tk and Tkinter are not bad, the developers will not find other alternatives. Let the developer give up the default choice, the toolkit must have other special things and functions. PyQt is this kind of toolkit. /6/ /15/

The advantages of PyQt are far beyond Tkinter. Qt and PyQt run fast. The design of Qt and PyQt is all object oriented. Qt supports a good designing widgets collection. It is bigger than Tk.

For the disadvantages, the license of Qt has been controlled more than other toolkit (At least on Non-Linux platform); to install Qt and PyQt correctly is complicated. The users of PyQt application need to try their best to finish the installation of Qt and PyQt, which makes the publishing of the application harder. /3/ /4/ /6/

PyQt is based on Qt license. Special thing is that it could be worked on UNIX/X11 platform's GPL, and work in Zaurus's Qt Palmtop Environment, and some older free Qt edition of MS Windows application package (Free-as-in-free-beer). The business license of PyQt is working on developing of MS Windows. /3/ /4/ /6/

In this work, the most important advantage of PyQt is that it is better than other toolkits and Qt uses a regime, which is called "signal/slots". It is among the widgets (and other objects) to send the messages and issues. This is the different than Tkinter and other toolkits which use "callback". Developers use the "signal/slots" in flexible and maintainable way to control the transfer message between objects. It is easier than "Callback" method. This advantage of Qt is clearer when the product become larger. /3/ /4/ /6/

3. INSTALLATION OF ENVIRONMENT

There should be a developing environment built in this work. The steps of installation are introduced in this part. In addition, the method of building “Hello world” testing application is introduced in this part.

3.1. Software requirement

The downloading step seems easy, but there are many editions in the Internet. The chosen of the correct installation version is an issue.

3.1.1. Python

The source links are easy to find on Python's website (<http://www.python.org/getit/>). The latest versions are Python 2.7.2 and Python 3.2.1.

In this work, Python 2.7.2 Windows Installer (Windows binary -- does not include source) was downloaded.

Download Python

The current production versions are [Python 2.7.2](#) and [Python 3.2.1](#).

Start with one of these versions for learning Python or if you want the most stability; they're both considered stable production releases.

If you don't know which version to use, start with Python 2.7; more existing third party software is compatible with Python 2 than Python 3 right now

For the MD5 checksums and OpenPGP signatures, look at the [detailed Python 2.7.2](#) page:

- [Python 2.7.2 Windows Installer](#) (Windows binary -- does not include source)
- [Python 2.7.2 Windows X86-64 Installer](#) (Windows AMD64 / Intel 64 / X86-64 binary [1] -- does not include source)
- [Python 2.7.2 Mac OS X 64-bit/32-bit x86-64/i386 Installer](#) (for Mac OS X 10.6 and 10.7 [2])
- [Python 2.7.2 Mac OS X 32-bit i386/PPC Installer](#) (for Mac OS X 10.3 through 10.6 [2])
- [Python 2.7.2 compressed source tarball](#) (for Linux, Unix or Mac OS X)
- [Python 2.7.2 bzipipped source tarball](#) (for Linux, Unix or Mac OS X, more compressed)

Also look at the [detailed Python 3.2.1](#) page:

- [Python 3.2.1 Windows x86 MSI Installer](#) (Windows binary -- does not include source)
- [Python 3.2.1 Windows X86-64 MSI Installer](#) (Windows AMD64 / Intel 64 / X86-64 binary [1] -- does not include source)
- [Python 3.2.1 Mac OS X 64-bit/32-bit x86-64/i386 Installer](#) (for Mac OS X 10.6 and 10.7 [2])
- [Python 3.2.1 Mac OS X 32-bit i386/PPC Installer](#) (for Mac OS X 10.3 through 10.6 [2])
- [Python 3.2.1 compressed source tarball](#) (for Linux, Unix or Mac OS X)
- [Python 3.2.1 bzipipped source tarball](#) (for Linux, Unix or Mac OS X, more compressed)

A [comprehensive list of all released versions](#) is available if you need source code for an older version of Python.

Fig.5. Python download /3.1.1, page7./

3.1.2. PyQt4

The installing application for PyQt4 library was downloaded from the link:
<http://www.riverbankcomputing.co.uk/static/Downloads/PyQt4/>

Index of /static/Downloads/PyQt4

	Name	Last modified	Size	Description
	Parent Directory	-		
	ChangeLog	02-Aug-2011 21:55	4.4K	
	ChangeLog-snapshot-4.8.6-e6842fdef3f9	26-Aug-2011 02:52	527	
	PyQt-Py2.5-x64-gpl-4.8.5-1.exe	02-Aug-2011 19:46	29M	
	PyQt-Py2.5-x86-gpl-4.8.5-1.exe	02-Aug-2011 19:56	26M	
	PyQt-Py2.6-x64-gpl-4.8.5-1.exe	02-Aug-2011 20:08	29M	
	PyQt-Py2.6-x86-gpl-4.8.5-1.exe	02-Aug-2011 20:18	26M	
	PyQt-Py2.7-x64-gpl-4.8.5-1.exe	02-Aug-2011 20:29	29M	
	PyQt-Py2.7-x86-gpl-4.8.5-1.exe	02-Aug-2011 20:38	26M	
	PyQt-Py3.1-x64-gpl-4.8.5-1.exe	02-Aug-2011 20:50	29M	
	PyQt-Py3.1-x86-gpl-4.8.5-1.exe	02-Aug-2011 21:00	26M	

Fig.6. PyQt4 download /3.1.2, page 8./

The Python2.7 package for MS Windows operating system was downloaded. For this reason, the “PyQt-Py2.7-x86-gpl-4.8.5-1.exe” application should be chosen.

3.1.3. Eric 4

The downloading link of Eric4 is

http://sourceforge.net/project/showfiles.php?group_id=119070&package_id=233329

Eric Integrated Development Environment by detlevo

Summary	Files	Reviews	Support	Develop
---------	-------	---------	---------	---------

Looking for the latest version? [Download eric5-5.1.4.zip \(8.5 MB\)](#)

Home / eric4 / stable










Name	Modified	Size
↑ Parent folder		
4.4.16	2011-07-03	
4.4.15	2011-06-02	
4.4.14	2011-05-08	
4.4.13	2011-04-02	
4.4.12	2011-02-07	
4.4.11	2011-01-02	
4.4.10	2010-12-04	
4.4.9	2010-10-09	
4.4.8	2010-09-05	

Fig.7. Eric4 download /3.1.3, page8./

The latest version is surely better. But for the purpose of the compile successful, the 4.2.2a package should be chosen. The operation system types also need to define. For example, the “zip” package is for MS Windows and “tar.gz” package for Linux/Unix.

3.2. The steps of installation

Step1. Install Python 2.7. The default path is “C:\python27”. After successful installation, it will be found on the start menu.

Step2. Install PyQt4. The default setting is the most directly method. It will be installed in “C:\python27” folder for defaulting.

Step3. Install Eric4. The “eric4-4.2.2a.zip” file should be extracted. Because it is developed by Python, the Python library should be installed before this step. After extracting, double click “install.py” file to start installing progress.

Step4. Double click on “eric4.bat” file to start Eric4. For the first time using, it needs to be configured. The “eric4-configure.bat” will be compiled, even if it does not be displayed automatically during the first time using.

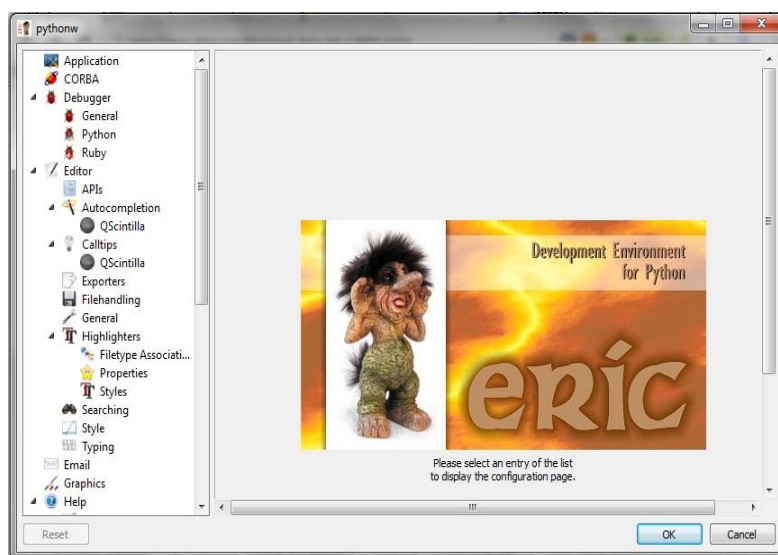


Fig.8. Eric4 Setting1 /3.2, page 10./

Choose “Autocompation” in “Edition” tag, make sure ”Autocompletion Enable”, ”Case sensitive” and ”Replace word” have been chosen.

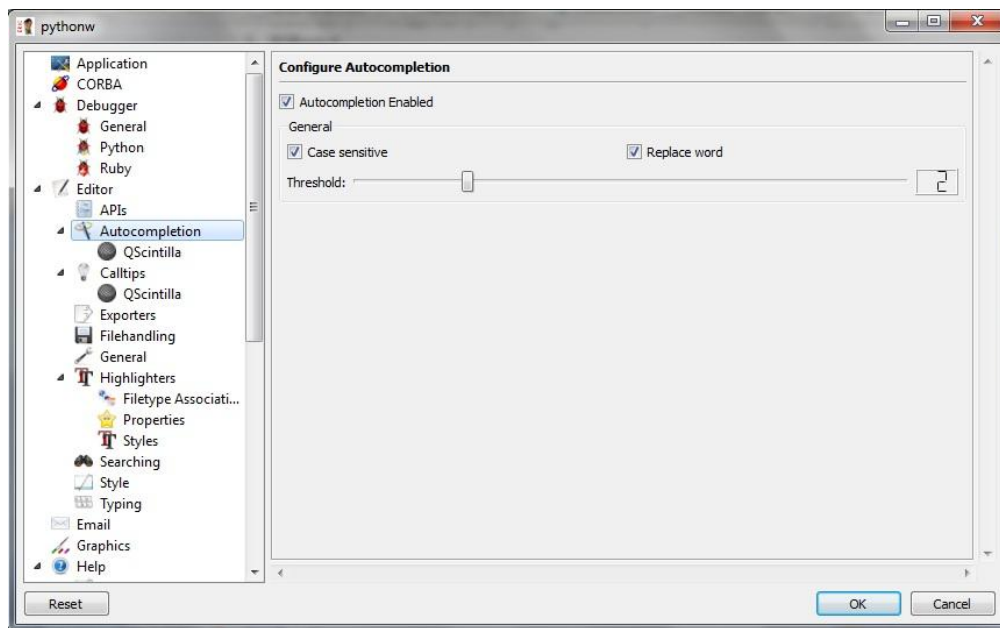


Fig.9. Eric Setting2 /3.2, page10./

In “QScintilla” tag, “Show single” and “Use fill-up characters” should be chose. In “Source” tag, the from “Document and API files” should be selected.

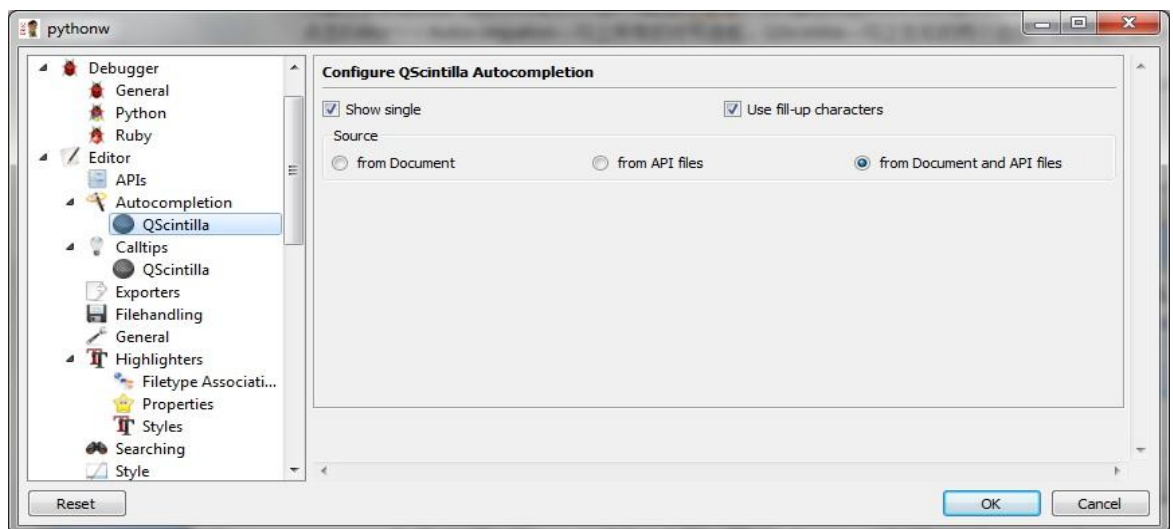


Fig.10. Eric Setting3 /3.2, page 10./

In “APIs” section of “Editor” tag, select “compile APIs Autocompation”. In “Language” combo-box, Python should be chose. Press button of “Add from installed APIs”, choose “eric4.api” file, and press “Compile APIs” button at last. After that, the installing is finished.

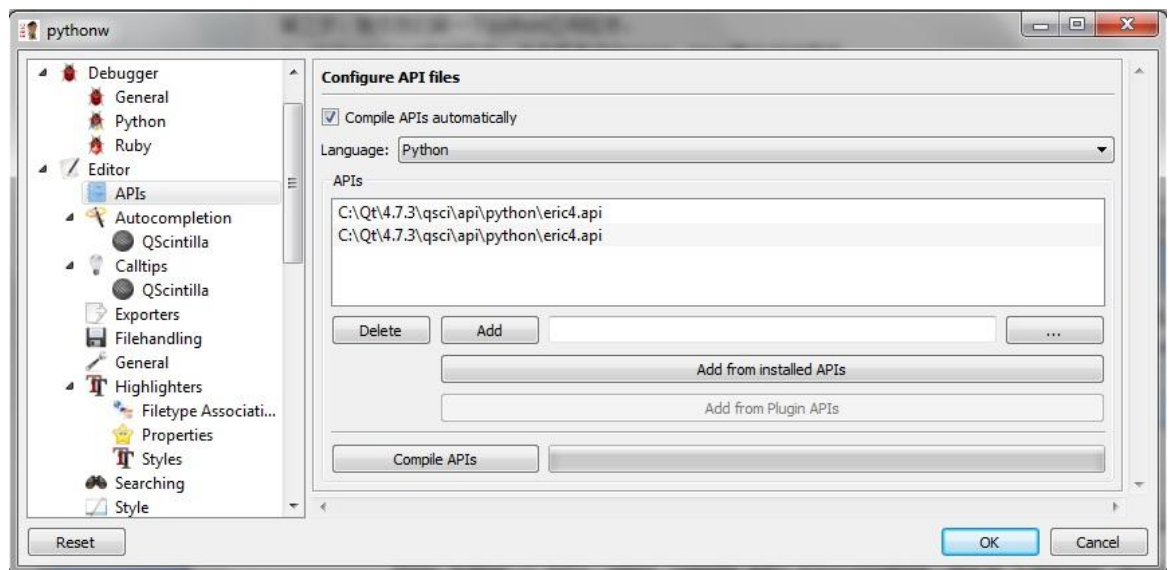


Fig.11. Eric4 Setting4 /3.2, page 11./

3.3. Hello World Application

Start Eric4 from “Eric4.bat” file, press “Project” button on the menu, then choose “New”. Set the data in the dialog window.

Project Name: HelloWorld

Project Type: QT4 GUI

Project Directory: Choose the directory where to save the project.

Different with other developing toolkits, the directory could be saved anywhere not in the same directory.

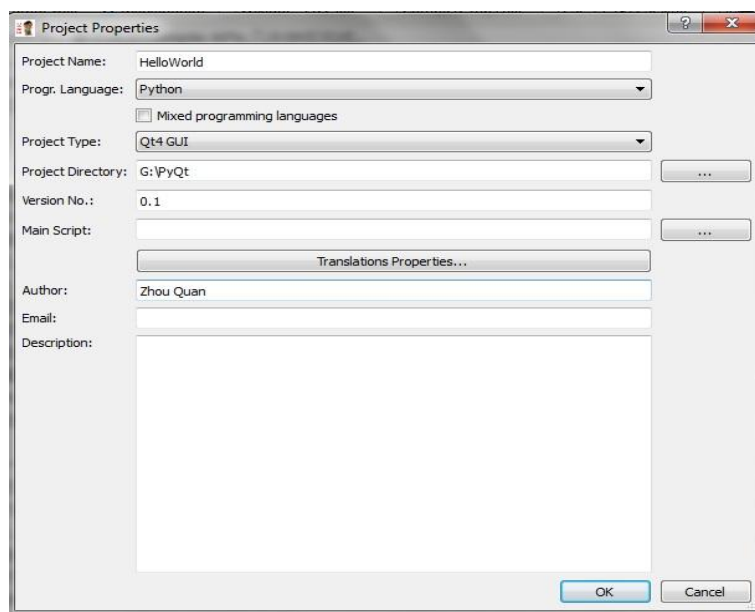


Fig.12. HelloWorld1 /3.3, page11./

Press OK. Then press “Yes” to add existing files to the project in the coming window.

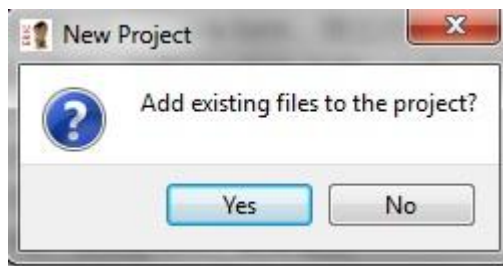


Fig.13. HelloWorld2 /3.3, page12./

Because there no need to specify the version controls system in this project, choose “None” in “the select version control system of the project” window.

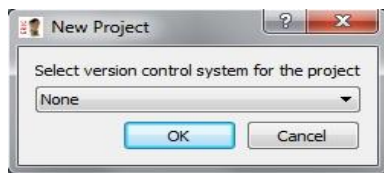


Fig14. HelloWorld3 /3.3, page12./

Click “Forms” section in “Project viewer” tag which is on the left-hand side of the application window. Right click of mouse in the blank space, and choose “New form” and choose the type to “dialog”.

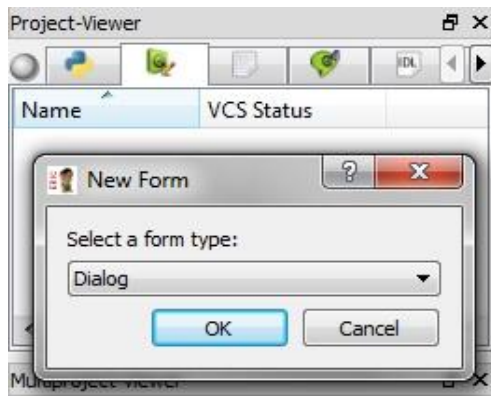


Fig.15. HelloWorld4 /3.3, page12./

Give a name to the UI file (“HelloWorld.ui” is chose in this project). After saving progress, the Qt Designer will be opened automatically.

For the design, it needs to add one “Line Editor”, and two “Push Button” in the UI. Change the object name of “Line Editor” to “LText”, and change the buttons’ name to “btn1” and “btn2”. Change the texts on the “btn1” to “Hello”, and “btn2” to “exit”. Press “+” button on the top of “Signal and Slot” window, choose sender to “btn2”, signal is “clicked()”, receiver is “Dialog”, slot is “close()”. As the image which is shown below.

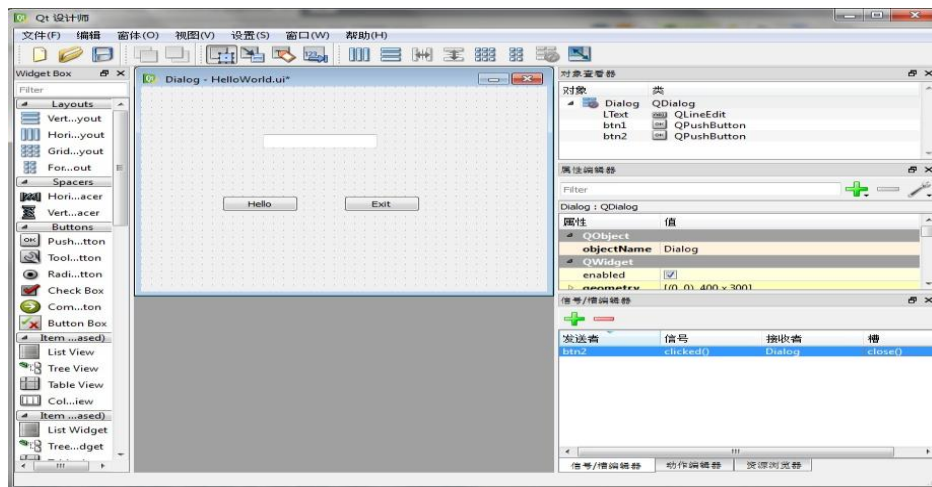


Fig.16. Qt Design /3.3, page 13./

Save all and return back to Eric4. There is a new file which is named “Ui_HelloWorld.py” in the source file. Double click on this files, press F2 to run it. A Python application has now been done.

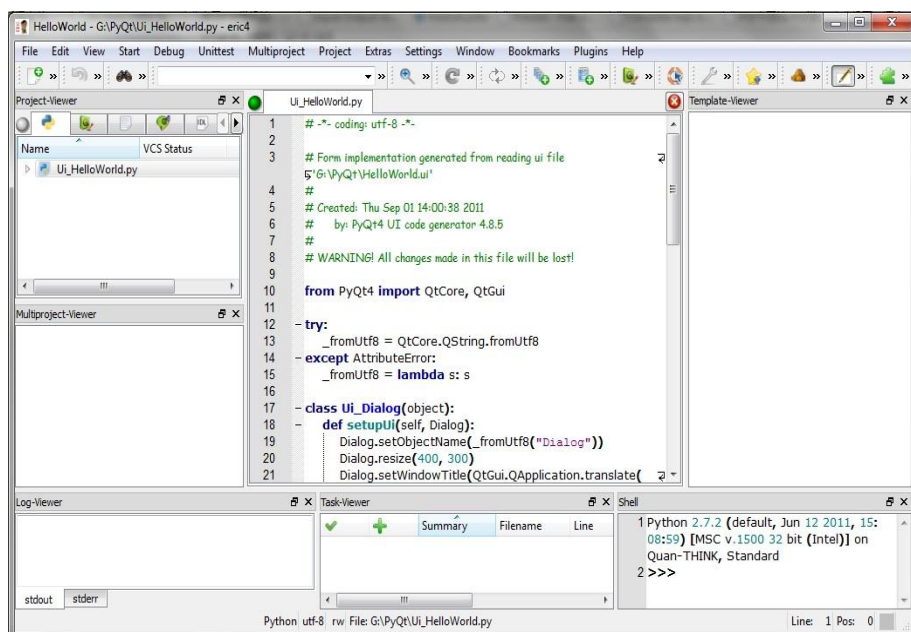


Fig.17. Eric4 /3.3, page 13./

Next, add the function on “btn1”. The purpose is after pressing “Hello” button, “hello world” will be displayed on the text line.

Right click “HelloWorld.ui”, choose “Generate Dialog Code”, select “ClassName”, and press “New”. Use the default section in this project. After that, choose “on_btn1_clicked()” for btn1, press ok.

Now, “HelloWorld.py” is displayed in the source file. Open this file and delete the code:

```
def on_btn1_clicked(self):  
    # TODO: not implemented yet  
    raise NotImplementedError
```

And using following code to instead:

```
def on_btn1_clicked(self):  
    self.LText.setText("Hello World")
```

Add

```
import PyQt4, PyQt4.QtGui, sys
```

on the top of the file.

And the ending should be

```
if __name__ == "__main__":  
    app = PyQt4.QtGui.QApplication(sys.argv)  
    dlg = Dialog()  
    dlg.show()  
    sys.exit(app.exec_())
```

Save and press F2 to run again. The product is working like the image below shows:

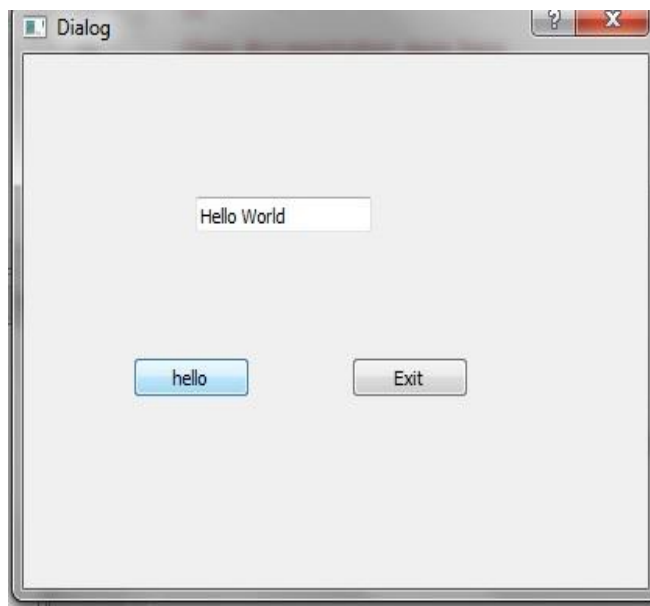


Fig.18. HelloWorld /3.3, page 14./

4. MINI WEB BROWSER

This part is introduced the way to develop a web browser application, which is using PyQt4 and Eric4 with Python language. This could be an example in the purpose of getting familiar with PyQy4 and Eric4.

4.1. Overview

The application should have an input blank for input the URL and a push button for connection to the internet server. From Qt library, there should be a function which is called QWebView to encode and display websites.

4.2. Implementation

First of all, create a project through eric4. Declare some data and information in "Project Properties" window.

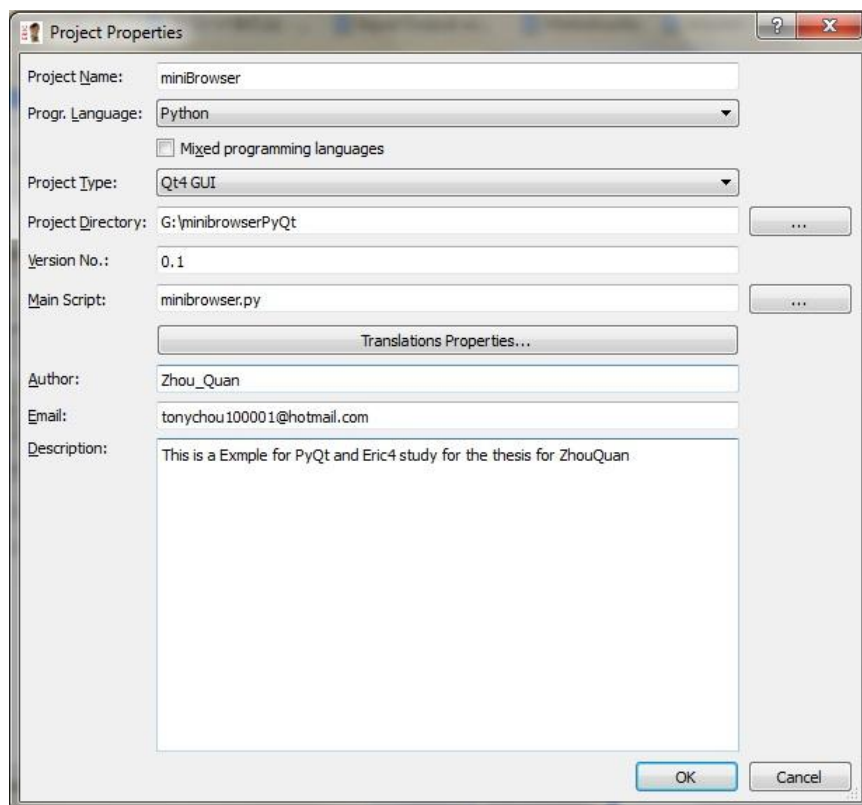


Fig.19. Example1.1 /4.2, page 15./

The coming dialog is asking whether there are needs to add existing files to the folder or not. Because it does not need any extra files, choose no. Choose “none” for the version control system for this project.

For the design, there should be a main window for the user interface. Create a new package for UI, the section can be found in “Project Viewer”->”Source”->right clicked.

Next, create the new form which is named/mainwindow.ui in “project viewer”->”form”->right clicked. Save it in the “ui” folder.

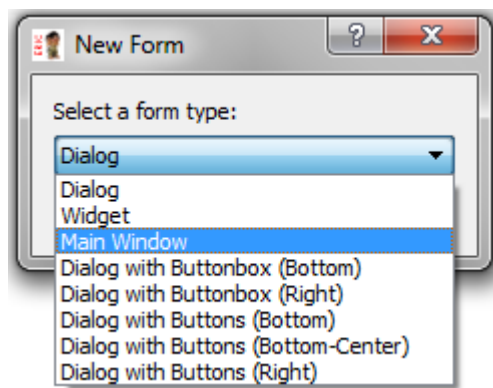


Fig.20. Example1.2 /4.2, page 16./

The Qt Designer is running. Add a QLineEdit and renamed to “txtUrl”, add QPushButton on the right of “txtUrl” and renamed to “btnNavigate”. Choose both of them, and right clicked the mouse, choose “Lay out” to “Lay Out Horizontally” in the menu.

Next, add QwebView widget. Right click in the blank area (Do not select any widgets), and choose “Lay out” to “Lay Out in a Grid”. That means, when the size of the window changes, the widgets in this user interface never change their lay out. The result should be like the image below.

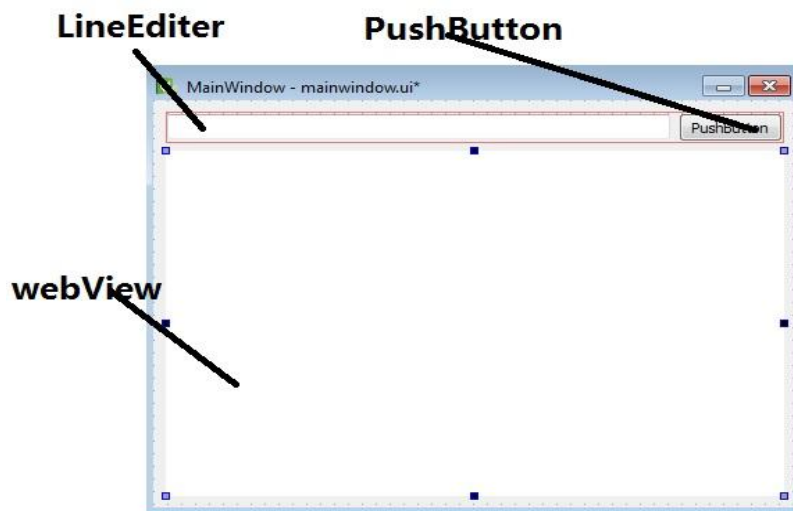


Fig.21. Example 1.3 /4.2, page 16./

Save all, and turn to Eric4. The UI design file should be displayed in “Project Viewer”->”Form”->”ui”, choose compile form in the right clicked menu. After the compile, choose “Generate Dialog Coding” still in the right clicked menu.

The UI design is finished. The next step is coding the functions for the web browser.

- A class which is called ‘mainwindow’ was created and the file named by ‘mainwindow.py’ to add the function codes.
- For the Navigate button, the application gets the URL from the txtUrl, also add “http://” in front of the URL.

```
@pyqtSignature("")
def on_btnNavigate_released(self):
    theUrl = self.txtUrl.text()
    if theUrl[0:7] != 'http://':
        theUrl = 'http://' + theUrl
    self.webView.setUrl(QUrl(theUrl))
```
- The title set for the main window.

```
@pyqtSignature("QString")
def on_webView_titleChanged(self, title):
    self.setWindowTitle(title)
```
- The URL set to the “webview” widget. After that, the website is displayed.

```
@pyqtSignature("QUrl")
def on_webView_urlChanged(self, url):
    self.txtUrl.setText(url.toString())
```
- The main class ‘minibrowser’ is added to run the application. That should be outside of “ui” folder. After that the code is added for this class.

```
from PyQt4 import QtCore, QtGui
from ui.mainwindow import MainWindow
if __name__ == "__main__":
    import sys
    app = QtGui.QApplication(sys.argv)
    ui = MainWindow()
    ui.show()
    sys.exit(app.exec_())
```

Save all files, and press F5 to run the application.

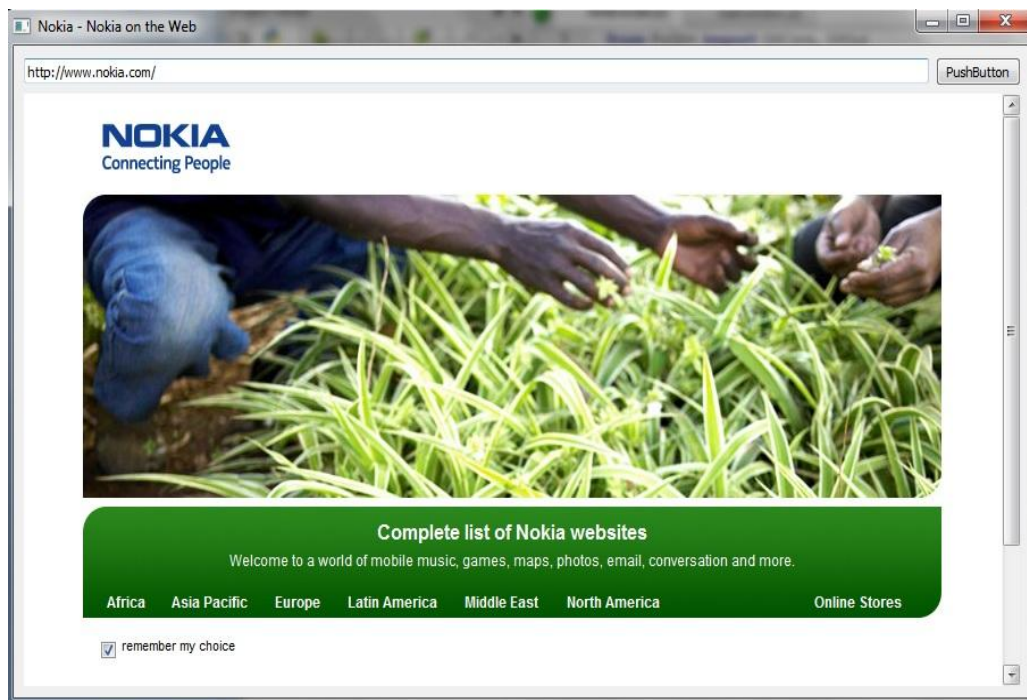


Fig.22. Example 1.4 /4.3, page 18./

5. ANALYSIS OF THE EXAMPLE

For the Python code analysis, there are two basic examples introduced in this part. The purpose of these examples is to get familiar with the Python language.

5.1. Basic input dialog

This example shows the way to use the standard input dialog. Qt framework supports a `QInputDialog` class, and this class conducts one simple dialog to get the user's input information. There are four types of data to input: string, int, double, and a combobox.

In this example, if a string input is needed, the `QlineEdit` is chosen; if the integer type or double type is needed, `QspinBox` is Chosen; if the combo-box is needed, the `QcomboBox` is chosen, and include input button (OK), and cancel input button (Cancel).



Fig.23. Example 2 /5.1, page 19./

5.1.1. Function of "slotName()"

For the example design, when the users press the button at the end of line "Name", the "slotName()" function is called.

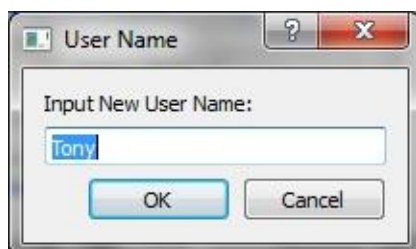


Fig.24. User Name /5.1.1, page 19./

```
def slotName(self):  
    name,ok=QInputDialog.getText(self,self.tr("User Name"),  
    self.tr("Input New"),
```



```
QLineEdit.Normal,self.nameLabel.text())
```

The function “getText()”(function) of “QInputDialog”(class) is called, and display the basic string input dialog. The standard syntax of ‘getText()’ is:

```
(QString, bool ok) QInputDialog.getText (QWidget, QString, QString, QLineEdit.EchoMode mode = QLineEdit.Normal, QString text = QString(), Qt.WindowFlags flags = 0)
```

The first variable is the parent window of standard input dialog; the second variable is the title of the dialog, the third is the notice; the fourth variable mode points the input type of QLineEdit in the dialog; and the fifth variable text is for the default string which displays in the QLineEdit; and the last variable shows the flag of the dialog.

In the third line of this function the value of “ok” estimates, if the user presses the OK button, the new input user name is displayed in the main window.

```
if ok and (not name.isEmpty()):
    self.nameLabel.setText(name)
```

5.1.2. Function of “slotSex()”

For the design, when the users press the button at the end of “Sex” line, the “slotSex()” function is called.

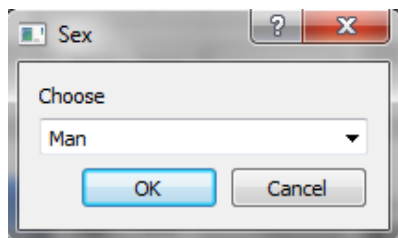


Fig.25. Combo box dialog /5.1.2, page 20./

These codes are created two objects for the combo-box, and make the data to QString.

```
list=QStringList()
list.append(self.tr("Man"))
list.append(self.tr("Woman"))
```

The “getItem()” function is called from the code of

“sex,ok=QInputDialog.getItem(self,self.tr("Sex"), self.tr("Choose"),list)” to the combo-box. And the standard syntax of “getItem()” is:

```
(QString, bool ok) getItem (QWidget, QString, QString, QStringList, int current = 0, bool editable = True, Qt.WindowFlags flags = 0)
```

The first variable points the parent window of the dialog, the second is title of the dialog, the third is the notice, the forth points the sections of “QComboBox” is a “QStringList” object, the fifth current display the number of the section in the combo box, when the dialog display the “QComboBox”, the sixth editable points the data of the combo box possible to edit or not, the last points the flag.

In the third line of this function the value of “ok” estimates, if the user presses the OK button, the new input user name is displayed in the main window.

```
if ok:
    self.sexLabel.setText(sex)
.
```

5.1.3. The function of “slotAge()”

For the design, when the users press the button at the end of “Age” line, the “slotAge()” function is called.

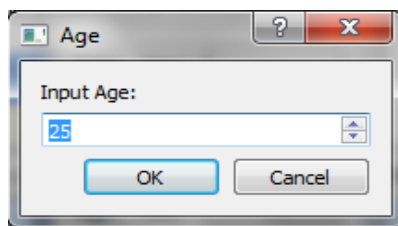


Fig.26. get integer dialog /5.1.3, page 21./

The function of “getInteger()” is called to get the integer data input. The standard syntax is:
(int, bool ok) getInteger (QWidget, QString, QString, int value = 0, int min = -2147483647, int max = 2147483647, int step = 1, Qt.WindowFlags flags = 0)

The first variable points the parent window, second is set the title of the dialog, the third is notice, the fourth is the default value of QSpinBox, the fifth and the sixth points the area of the input value of the QSpinBox, and the seventh step points the change value of QSpinBox.

5.1.4. Function of “slotStature()”

For the design, when the users press the button at the end of “High” line, the “slotStature()” function will be called.



Fig.27. Get double dialog /5.1.4, page 21./

The function of “getDouble ()” is called to get the integer data input. The standard syntax is:

(float, bool ok) getDouble (QWidget, QString, QString, float value = 0, float min = -2147483647, float max = 2147483647, int decimals = 1, Qt.WindowFlags flags = 0)

The first variable points the parent window, second is set the title of the dialog, the third is notice, the forth is the default value of "QspinBox", the fifth and the sixth point the area of the input value of the "QspinBox", and the seventh decimals points the change value of "Qspinbox".

5.2. Progress Bar

Qt framework supplies two ways to display the progress bar: one is QProgressBar, the other is QProgressDialog. A horizontal or vertical method is supported by QProgressBar to display the progress bar for presenting the progress. A slow speed progress display method is supported by QProgressDialog to present the progress.

QProgressBar has some important property value: minimum, maximum decide the minimum and maximum value of the progress bar to display, and format decides the display word's format. There are three formats to display: %p%, %v, %m. '%p%' displays the percentage which has been finished. This is the default method; '%v' displays the current progress; '%m' displays the progressive value of the whole event. InvertedAppearance property could make the progress display in the other direction.

QProgressDialog has also some important property values deciding when the progress dialog displays, and the length of time for the displaying. They are: minimum, maximum, and minimumDuration. The 'minimum' and 'maximum' are the min-value and max-value of the progress bar display, deciding the range of variation; 'minimumDuration' is the waiting time before the progress dialog displays. The system calculates the time of the whole event. If the time is longer than the setting time 'minimumDuration', the dialog is displayed. If shorter, the dialog is not displayed.

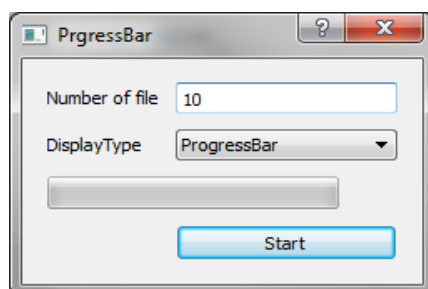


Fig.28. Progress Bar /5.2, page 22./

The function is for the "start" button. After pressing, the progress bar will be started in the selected type.

5.2.1. Using progress bar

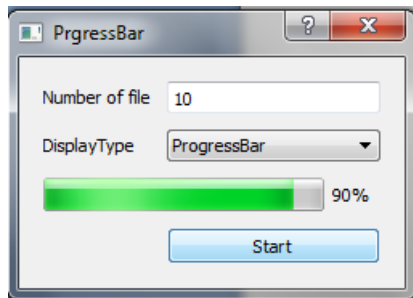


Fig.29. Prograss Bar 1 /5.2.1, page 23./

The codes of `num=int(self.numLineEdit.text())` are used to get the number of files which will be copied in this event, it is the progressive value of the event.

```
if self.typeComboBox.currentIndex()==0:
    self.progressBar.setMinimum(0)
    self.progressBar.setMaximum(num)
```

These codes are used to set the range of the progress value, from 0 to “num” files to copy.

```
for i in range(num):
    self.progressBar.setValue(i)
    QThread.msleep(100)
```

Each copying file progress is simulated. Through “Qthread.msleep(100)”, the value in the progress bar is the number of copied files. Once the file copy process is finished, the value adds one.

5.2.2. Using progress dialog

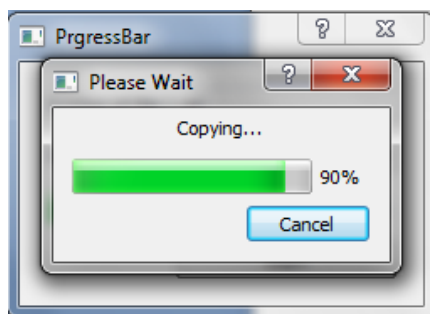


Fig.30. Progress Dialog /5.2.2, page 23./

```
elif self.typeComboBox.currentIndex()==1:
    progressDialog=QProgressDialog(self)
```

A progress dialog is created by these codes.

```
progressDialog.setWindowModality(Qt.WindowModal)
```

```
progressDialog.setMinimumDuration(5)
```

These codes set the modal way how the progress dialog is displayed. That means, when the dialog works, another window does not get any input signal. The waiting time is set to 5 seconds.

```
progressDialog.setWindowTitle(self.tr("Please Wait"))  
progressDialog.setLabelText(self.tr("Copying..."))  
progressDialog.setCancelButtonText(self.tr("Cancel"))
```

The title and texts on the dialog window is set by these codes.

```
progressDialog.setRange(0,num)
```

The range of the progressive value for the progress dialog is set.

```
for i in range(num):  
progressDialog.setValue(i)  
QThread.msleep(100)
```

Each copying file progress is simulated. Through “Qthread.msleep(100)”, the value in the progress bar is the number of the copied files. After one file copying process is finished, the value adds one.

```
if progressDialog.wasCanceled():  
return
```

If the “cancel” button is clicked, the dialog and the loop are closed. But there are some cleaning functions to be done in the real product.

The using of the progress dialog has two ways: modal way and un-modal way. In this example, the modal way is used. The modal way is easy to use, but must the progressEvents must be used to make the event loop working well. If using un-modal way, the setting time progress value must be implemented through QTimer.

6. PACKAGE

The next step of developing is packing the source code for the mobile devices. Because Maemo platform is Linux, the Linux operating system is needed in this step. In this project, the Oracle VM virtual box is used, and Maemo5 SDK in ScratchBox is used./10/ /11/

Step1. Create an empty folder for programming

An empty folder should be created in home folder of Linux. The version name should be included in the folder name, for example minibrowser-1.0.

Step2. Use “dh_make createbasic” to control files

Translate path to the minibrowser-1.0 from terminal, and type the command:

```
DEBFULLNAME="Cute Qt" dh_make --createorig --single -e cuteqt@cuteqt.com -c gpl
```

Install dh-make package if the system do not install it. In this command, ‘**DEBFULLNAME**’ is decide the name of the publisher, ‘-single’ means that it is an application, ‘-e’ means the email address for the developer, ‘-c’ is the publishing type of the application, it is GPL in here. After working, a new folder called Debian has appeared in the minibrowser-1.0 folder. In this package, there are some useful files which will be used for packaging.

Step3. The sources are copied in the directory

All files and folders which have been developed are copied into the folder of /hom/minibrowser-1.0/minibrowser.

Step4. A file called “top.pro” is created in minibrowser-1.0 folder

The code of the file should be:

```
QMAKEVERSION = $$[QMAKE_VERSION]
ISQT4 = $$find(QMAKEVERSION, ^[2-9])
isEmpty( ISQT4 ) {
    error("Use the qmake include with Qt4.4 or greater, on Debian that is
    qmake-qt4");
}
TEMPLATE = subdirs
SUBDIRS = minibrowser
```

First of all, the code “qmake” is used to test if the version of Qt is four or not. The source folder is pointed by “subdirs”.

Step5. The “minibrowser.pro” is edited

Some variable for INSTALLS are modified in this step. It is decided by the files which should be packed and the path of mobile devices when the application has been installed.

```
target.path = $$PREFIX/bin
desktop.files = wiggly.desktop
desktop.path = /usr/share/application/hildon/
INSTALLS += target desktop
```

Two variables have been selected, the application and the '.desk' file. '.desk' file is the entrance to find the application when the Maemo system runs the application; it includes the source file and icon, displays name, path and so on. The '.desktop' paths of Maemo are in /usr/share/application/hidon. '.pro' declares which files should be packed; 'path' declares the target path when it is installed in a new system.

Step6.A new “minibrowser.desktop” file is added in “minibrowser” folder

```
[Desktop Entry]
Encoding=UTF-8
Version=1.0
Type=Application
Name= minibrowser
Exec=/opt/usr/bin/minibrowser
Icon=myapp
X-HildonDesk-ShowInToolbar=true
X-Osso-Type=application/x-executable
```

The most important variable is 'Exec'; it declares the path of the application. If there is any error, the application cannot work.

Step7. Rules file is edited

In the minibrowser-1.0/debian folder, the rules file declares the operations when packing. The main difference is “./configure && make” for normal. But in Qt, it should be 'qmake && make'.

Table.1. Comparison of the modify rule file

Befor edit	After edit
<pre>configure: configure-stamp configure-stamp: dh_testdir # Add here commands to configure the package. touch configure-stamp build-stamp: configure-stamp dh_testdir # Add here commands to compile the package. \$(MAKE) #docbook-to-man debian/minibrowser.sgml > minibrowser.1 touch \$@ build: build-stamp</pre>	<pre>builddir: mkdir -p builddir builddir/Makefile:builddir cd builddir && qmakePREFIX=/opt/usr../top.pro build-stamp: builddir/Makefile dh_testdir # Add here commands to compile the package. cd builddir && \$(MAKE) #docbook-to-man debian/ minibrowser.sgml > minibrowser.1 touch \$@ build: build-stamp</pre>

From the table above (Table.1), the new file creates a new “builddir” compile directory, and “qmake” is used to create “Makefile”.

Because the new rules are added to the “builddir” compile individually, “make clean” and “make install” have some differences. It is shown in the table below (Table.2):

Table.2. Comparison of modify Make file

Befor edit	After edit
<pre>clean: dh_testdir dh_testroot rm -f build-stamp configure-stamp# Add here commands to clean up after the build process. -\$(MAKE) clean dh_clean install: build dh_testdir dh_testroot dh_clean -k dh_installdirs \$(MAKE) DESTDIR=\$(CURDIR)/debian/minibrowse r install</pre>	<pre>clean: dh_testdir dh_testroot rm -f build-stamp configure-stamp # Add here commands to clean up after the build process. #-\$(MAKE) clean rm -rf builddir dh_cleaninstall: build dh_testdir dh_testroot dh_clean -k dh_installdirs cd builddir && \$(MAKE) INSTALL_ROOT=\$(CURDIR)/debian/minib rowser install</pre>

Step8. Package

Turn to minibrowser-1.0 from terminal, and type command:

```
Dpkg-buildpackage -rfakeroot -b
```

There is a “.deb” file is created in the upper folder.

7. CONCLUSIONS

Python language is a new programming language for me, and it is not one of the main developing languages nowadays. There are not many sources which can be found on the websites. Maemo is not as a popular mobile platform as Android. The method of using Python in Maemo platform is a great issue. Nokia seems to give up the developing on Maemo, and the new MeeGo is just on the market (Nokia N9-00) now. That makes the source for Maemo limited. For this thesis, I started from the language study by reading the tutoring document for Python, and tried to implement some examples and simple applications.

The Maemo study followed the pre-study. I have learned the Mobile Programming during my study period, so there is not much trouble in using C++ language to do the programming for Maemo. I found that the Python language is easy to use, but it cannot be used in Qt Creator directly. Selecting another IDE to develop Python in Qt framework was another challenge for me. I thought Maemo supports Qt framework, and the Qt creator can be used in developing work of Maemo. So I started to use Python in Qt framework. If it is done, and packed into the Maemo form, the application could work on Maemo platform. After reading hundreds of documents for Python IDE, I chose PyQt4 and Eric4 for corporation.

After programming, an interesting issue - pack- should be solved. Because the application will be used on real mobile devices, the hardware is quite different from computer. Because the installing application is different from a normal Debian application which is used on the computer. Through research the method was found. I could not get the real Maemo devices, Nokia N900, so I could not test the result in the end. In my opinion, it is not successful ending for me.

Through the thesis writing, I have improved my language a lot. I have written lots of project reports before, but this time I needed to read the document without the teacher's help. My vocabulary in the professional area has increased. Writing in a professional way is really good for me.

The latest news, a new operating system which is based on Linux will be published. It is named TIZEN. It is an open source system like others which are based on Linux, and for mobile devices (Smartphone, netbook, tablets, smart TV, and invehicle infotainment devices). It is a product from the corporation of Intel and Samsung. Somebody says Meego was dead, but I do not agree. In my opinion, Meego has the advantages for itself. Because the SDK of TIZEN is not published yet, I cannot say whether it is better or not. But Meego has been developed for so many years, and it is supported by Qt framework, that means there will be lots of third-part applications for the customer. Like Maemo 5 devices (Nokia N900), somebody think that it is an unsuccessful product, but it is still a good device for the professional customers.

In this thesis, the developing method is not just for Maemo or MeeGo. It can be used in all applications which use Python language and Qt Framework.

8. REFERENCES

- /1/ Nokia furom, wiki, Python [www.document]
[<http://www.developer.nokia.com/Community/Wiki/Category:Python>] 12 October. 2010.
- /2/ Swaroop CH, Python tutoring book, a byte of python copyright 2003-2005
[www.document]
[http://woodpecker.org.cn/abyteofpython_cn/chinese/ch01s02.html] 15 December. 2010
- /3/ Python beginners Guide [www.document]
[<http://wiki.python.org/moin/BeginnersGuide/Overview>] 3 January. 2011.
- /4/ First programs in PyQt4 toolkit [www.document]
[<http://zetcode.com/tutorials/pyqt4/firstprograms/>] 6 March. 2011.
- /5/ Rob Oakes, Installation of PyQt on Windows, May 12, 2009[www.document]
[<http://www.oak-tree.us/blog/index.php/2009/05/12/pyqt-windows>] 15 March. 2011
- /6/ PyQt 4.8.5 Reference Guide [www.document]
[<http://www.riverbankcomputing.co.uk/static/Docs/PyQt4/html/installation.html>] 20 March. 2011.
- /7/ Sebastian Montabone, Create first application for Nokia N900, [www.documnet]
[<http://www.samontab.com/web/2011/01/how-to-create-your-first-application-for-the-nokia-n900-using-qt/>] 26 March. 2011.
- /8/ mikec, Qt Designer/Python for Windows, [www.document]
[<http://talk.maemo.org/showthread.php?t=43663>] 31 March. 2011
- /9/ Maemo.org community [www.documnet]
[<http://maemo.org/community/>] 3 September. 2010
- /10 /PyMaemo community [www.document]
[<http://wiki.maemo.org/PyMaemo>] 4 May. 2011.
- /11/ CuteQt Blog [www.document]
[<http://www.cuteqt.com/blog/?p=1449>] 4 May. 2011.
- /12/ PyQt4 Example Analyze, standard dialog, [www.document]
[<http://blog.csdn.net/chumpklutz/article/details/6052115>] 4 May. 2011.
- /13/ PyQt4 Example Analyze, progress bar, [www.document]
[<http://blog.csdn.net/chumpklutz/article/details/6059473>] 6 May. 2011.
- /14/ The Eric Python IDE[www.document]
[<http://eric-ide.python-projects.org/>] 5 May. 2011.

/15/ Qt official web site [www.document]
[<http://qt.nokia.com/products/>] 4 September. 2011.